МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
“ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

МЕТОДИЧНІ ВКАЗІВКИ

до виконання контрольних робіт за курсом “ Основи комп’ютерної інженерії”
для студентів заочної форми навчання
спеціальності “Комп’ютерна інженерія”

Затверджено
редакційно-видавничою
радою НТУ “ХПІ”,
протокол № 4 від 03.07.2020 р.

Харків НТУ “ХПІ” 2020

УДК 004.7

Рецензенти:
професор кафедри «Обчислювальна техніка та програмування»
Філоненко А.М.

Методичні вказівки до виконання контрольних робіт за курсом “ Основи комп’ютерної інженерії” для студентів заочної форми навчання спеціальності “Комп’ютерна інженерія” /Уклад: Гавриленко С.Ю. – Харків: НТУ “ХПІ”, 2020. – 37 с.

Укладачі: С.Ю. Гавриленко, О.В. Октябрьова
Кафедра обчислювальної техніки та програмування

УДК 004.4’22; 004.4’24; 519.685

ЗМІСТ

ВСТУП

З появою обчислювальної техніки з’явилась необхідність використовувати різні системи числення.

Усі сучасні комп’ютери мають досить розвинуту систему команд, що включає десятки і сотні машинних операцій. Однак виконання будь-якої операції засновано на використанні найпростіших мікрооперацій типу додавання і зсуву. Це дозволяє мати єдиний арифметико-логічний пристрій для виконання будь-яких операцій, які пов’язані з обробкою інформації.

У всіх комп’ютерів без винятку всі операції виконуються над числами, поданими спеціальними машинними кодами. Їхнє використання дозволяє обробляти знакові розряди чисел так само, як і значущі розряди, а також заміняти операцію віднімання операцією додавання.

Найбільш поширені чотири системи числення – двійкова, вісімкова, десяткова і шістнадцяткова.

1 КОНТРОЛЬНА РОБОТА 1

1.1 Завдання до контрольної роботи 1

Завдання до контрольної роботи 1 містить 5 пунктів та наведено нижче.

1. Вибрати варіант завдання та визначити за табл. 1.1 вихідні дані A_{10} і B_{10}.

Номер варіанта: (i, j) – це перша і друга цифри двозначного числа, що визначається як сума: порядковий номер в журналі групи + останні дві цифри року вступу + індекс групи (0– якщо індекс відсутній, 1 – "а", 2 – "б", 3 – "в", 4 – "г", ", 5 – "д" і т.д.).

Наприклад, для студента з номером 7 у групі КІТ-1**19**б номер варіанта буде таким: 7+19+2 = 29 (i = 2, j = 9).

Отже, студент повинен вибрати з таблиці 1 число на перетині стовпця 2 та рядка 0. Таким чином, A_{10} = 13,4.

Визначити число B, розташоване на перетині i+1-стовпця та j-рядка. У нашому прикладі B_{10} = 43,3.

Таблиця 1.1 – Вихідні дані

	i = 1	i = 2	i = 3	i = 4	i = 5	i = 6	i = 7	i = 8	i = 9
j = 0	19,5	37,5	16,6	73,9	76,3	45,5	57,3	64,7	23,8
j = 1	49,8	45,7	13,6	83,3	67,4	19,1	78,4	32,3	36,3
j = 2	22,6	53,1	24,5	66,3	78,1	65,7	89,3	15,4	**13,4**
j = 3	57,7	74,4	15,3	38,4	56,3	29,4	68,1	54,3	**43,3**
j = 4	25,2	51,1	56,3	78,5	67,2	48,4	27,6	37,8	28,4
j = 5	54,9	78,3	35,9	17,4	16,4	44,7	78,9	25,2	94,5
j = 6	97,6	15,2	63,2	69,3	89,2	59,4	18,7	67,4	61,2
j = 7	82,4	43,3	74,8	24,7	48,3	15,3	75,5	12,5	35,8
j = 8	24,7	23,4	75,9	61,3	14,3	35,6	23,6	46,6	15,7
j = 9	36,1	48,5	24,6	75,3	36,5	55,7	62,1	45,2	35,4

1.1. Вхідні числа A і B, задані в десятковому коді, подати в двійковому коді з точністю до п'яти знаків після коми.

1.2. Округлити числа *A* і *B* кожне до найближчого цілого числа і вважати їх далі відповідно цілими числами *C* та *D*. Останні треба подати в кодах: двійковому, вісімковому, шістнадцятковому і двійково-десятковому.

2. Подати число *C* в однобайтовому полі. В числі *C* визначити 3-й біт, інвертувати 5-й біт та встановити 4-й біт в 1, а 1-й біт – в 0. Виконати лінійний логічний зсув числа *C* на *i* розрядів вліво, та циклічний арифметичний зсув числа *C* на *i* розрядів вправо.

3. Подати двійкові числа *C* і *D* у прямому, зворотному і додатковому модифікованому кодах у форматі з фіксованою комою.

4. Над отриманими числами *C* і *D* виконати операції додавання в зворотному і додатковому кодах (використовуючи по черзі знаки "+" і "–" перед кожним з цих чисел):

$D+C$ $D+(-C)$ $(-D)+C$ $-D+(-C)$

5. Подати числа *A* та *B* у форматі з плаваючою комою. Знайти $F = A+B$. Результат обчислень подати в прямому коді, а потім перевести в десятковий код та виконати перевірку.

1.2. Форми подання чисел у різних системах числення

Системою числення називається спосіб зображення чисел за допомогою обмеженого набору символів, що мають визначені кількісні значення.

Систему числення утворює сукупність правил і прийомів подання чисел за допомогою набору знаків (цифр).

Подання чисел у різних системах числення допускає однозначне перетворення їх з однієї системи в іншу. В комп’ютері конвертація з однієї системи числення в іншу здійснюється автоматично за допомогою спеціальних програм. Правила конвертації цілих і дійсних чисел відрізняються.

Розрізняють позиційні і непозиційні системи числення. У позиційних системах кожна цифра числа має визначену вагу, що залежить від позиції цифри в послідовності, що зображує число. Позиція цифри називається розрядом. У позиційній системі числення будь-яке число можна подати у вигляді

$$A = a_{m-1}a_{m-2}\ldots a_i a_0\,,\, a_{-1}a_{-2}\ldots a_{-k} = a_{m-1}*N^{m-1}+a_{m-2}*N^{m-2}+\ldots+a_{-k}*N^{-k} \qquad (1.1)$$

$$\text{тобто} \quad A = \sum_{i=-k}^{m-1} a_i * N^i$$

де a_i – i-а цифра числа; k – кількість цифр у дробовій частині числа; m – кількість цифр у цілій частині числа; N – основа системи числення.

Основа системи числення N показує, у скільки разів "вага" i-го розряду більше "ваги" *(i–1)*-го розряду. Ціла частина числа відокремлюється від дробової частини крапкою (комою).

Ціла частина числа A_{N1}, з основою N_1, переводиться в систему числення з основою $N2$ шляхом послідовного ділення цілої частини числа A_{N1} на записану у вигляді числа з основою $N1$ основу $N2$, до одержання залишку. Отримана частка знову ділиться на основу $N2$, і цей процес треба повторювати доти, доки частка не стане менше дільника. Отримані залишки від ділення й остання частка записуються в порядку, *зворотному* отриманому при діленні. Сформоване число і буде цілим числом з основою $N2$.

Дробова частина числа A_{N1}, з основою $N1$, переводиться в систему числення з основою $N2$ шляхом послідовного множення дробової частини числа A_{N1} на основу $N2$, записану у вигляді числа з основою $N1$. При кожному множенні ціла частина добутку береться у вигляді чергової цифри відповідного розряду, а дробова частина, що залишилася, приймається за нове множене. Число множень визначає розрядність отриманого результату, що представляє дробову частину числа A_{N1} у системі числення $N2$. Дробова частина числа при переводі часто представляється неточно.

В усіх сучасних комп’ютерах для подання числової інформації використовується двійкова система числення, в якій число зображується за допомогою цифр $\{0,1\}$. Це обумовлено:

- більш простою реалізацією алгоритмів виконання арифметичних і логічних операцій;
- більш надійною фізичною реалізацією основних функцій, тому що в них мають місце усього два стани (0 і 1);
- економічністю апаратурної реалізації всіх схем комп’ютера.

У шістнадцятковій системі числення для подання числа передбачені цифри та букви – {0, 1, 2, ..., 9, *A*, *B*, *C*, *D*, *E*, *F*}, де буквою *A* позначається "цифра" 10, буквою *B* – "цифра" 11, ..., буквою *F* – "цифра" 15.

У вісімковій системі числення для подання числа передбачені вісім цифр – {0, 1, 2, 3, 4, 5, 6, 7}. Вона широко використовується в багатьох спеціалізованих комп'ютерів .

Вісімкова і шістнадцяткова системи числення є похідними від двійкової системи, тому що $16 = 2^4$, а $8 = 2^3$. Вони використовуються в основному для більш компактного подання двійкової інформації, тому що запис значення чисел досягається істотно меншим числом цифр.

У двійково-десятковій системі числення кожній десятковій цифрі відповідає особиста сукупність чотирьох бітів у двійковій системі числення.

Якщо число *A,* записане в двійковій системі числення, розбити на тріади (по 3 біта) і кожну тріаду записати в вісімковій системі числення, то ми отримаємо запис числа *A* у вісімковій системі числення.

Якщо число *A*, записане в двійковій системі числення, розбити на тетради (по 4 біта) і кожну тетраду записати в шістнадцятковій системі числення, то ми отримаємо запис числа *A* в шістнадцятковій системі числення.

Кожний розряд числа, записаного в *N*-ковій системі числення, має свою вагу. Наприклад, вага розрядів для двійкової системи числення наступна:

Номери бітів	7	6	5	4	3	2	1	0
Вага розрядів	128	64	32	16	8	4	2	1

Приклад 1.

Маємо числа $A = 37{,}5$ і $B = 16{,}6$.

Переведемо ці числа у двійкову систему числення. Для цього цілу частину кожного з них ділимо на основу 2 двійкової системи числення, а дробову частину кожного з них послідовно множимо на основу 2 двійкової системи числення.

Представимо число *A* в двійковій системі числення:

37|2
36| 18|2
1 18| 9|2
0 8 | 4|2
1 4| 2|2
0 2| 1
0

0,5
2
1),00 – виноситься одиниця

Для отримання п’яти знаків після крапки дописуємо незначущі нулі справа. Таким чином, A_2 = 100101.10000

Виконаємо перевірку цілої частини отриманого числа:

Номери бітів	8	7	6	5	4	3	2	1	0
Вага розрядів	2^8 = 256	2^7 = 128	2^6 = 64	2^5 = 32	2^4 = 16	2^3 = 8	2^2 = 4	2^1 = 2	2^0 = 1
A	0	0	0	1	0	0	1	0	1

Згідно з формулою (1) визначаємо десяткове значення цілої частини числа A:

$A = 0*2^8 + 0*2^7 + 0*2^6 + 1*2^5 + 0*2^4 + 0*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = 37$

Виконаємо перевірку дробової частини отриманого числа:

Номери бітів	1	2	3	4	5
Вага розрядів	2^{-1} = 0,5	2^{-2} = 0,250	2^{-3} = 0,125	2^{-4} = 0,0625	2^{-5} = 0,03125
A	1	0	0	0	0

Згідно з формулою (1) визначаємо десяткове значення дробової частини числа A:

$A = 1*2^{-1} + 0*2^{-2} + 0*2^{-3} + 0*2^{-4} + 0*2^{-5} = 0.5$

Таким чином, визначити десяткове значення числа *A* можна за рахунок складення ваги тих двійкових розрядів, у яких значення розряду дорівнює 1, тобто A_{10} = 32+4+1+0.5 = 37.5

Аналогічно переведемо в двійкову систему числення число *B* та виконаємо перевірку:

16|2	0,6
16| 8|2	2
0 8| 4|2	1),2 виноситься одиниця
0 4| 2|2	2
0 2| 1	0),4 виноситься нуль
0	2
	0),8 виноситься нуль
	2
	1),6 виноситься одиниця
	2
	1),2 виноситься одиниця

Кількість множень обмежується точністю підрахунку (в нашому випадку – п’ять розрядів). Таким чином, B_2 = 10000.10011.

Згідно з формулою (1) визначаємо десяткове значення числа *B*:

$B_{10} = 1*2^4 + 0*2^3 + 0*2^2 + 0*2^1 + 0*2^0 + 1*2^{-1} + 0*2^{-2} + 0*2^{-3} + 1*2^{-4} + 1*2^{-5} = 16.59375$

Округлимо число *A* та отримаємо число C_{10}=38. Округлимо число *B* та отримаємо число D_{10}=17. Запишемо *C* та *D* у двійковій, вісімковій, шістнадцятковій і двійково-десятковій системах числення. Для переходу в вісімкову систему числення розбиваємо число на тріади (по 3 біта) і кожну тріаду записуємо в десятковій системі числення.

Для переходу в шістнадцяткову систему числення розбиваємо число на тетради (по 4 біта) і кожну тетраду записуємо в десятковій системі числення.

C_{10} = 38	D_{10} = 17
C_2 = 100110	D_2 = 10001
$C_2 = \underbrace{100}\ \underbrace{110}$	$D_2 = \underbrace{10}\ \underbrace{001}$
C_8= 4 6	D_8 = 2 1

$C_2 = \underbrace{10}_{2}\ \underbrace{0110}_{6}$

$C_h = 26$

$C_{2\text{-}10} = 0011\ 1000$

$D_2 = \underbrace{1}\ \underbrace{0001}$

$D_8 = 1\ \ 1$

$D_h = 11$

$D_{2\text{-}10} = 0001\ 0111$

Для переходу в вісімкову та шістнадцяткову системи числення десяткове число також можна розділити на основу системи числення 8 або 16 відповідно. Наприклад, для числа *D* будемо мати:

$$\begin{array}{r|l} 17 & \underline{8} \\ \underline{16} & 2 \\ 1 & \end{array} \qquad \begin{array}{r|l} 17 & \underline{16} \\ \underline{16} & 1 \\ 1 & \end{array} \qquad \begin{array}{l} D_8 = 21 \\ D_h = 11 \end{array}$$

1.3 Побітові логічні операції

Логічні команди в основному використовуються для маніпулювання двійковими значеннями. Логічні команди названі логічними тому, що вони діють за правилами формальної логіки, а не арифметики. Вони змінюють значення бітів у регістрах чи комірках пам’яті.

Команди маніпулювання бітами розділені на три групи: логічні команди, команди зсуву і команди циклічного зсуву.

Найбільш поширеними логічними операціями є: інвертування (NOT), операція логічного складання “АБО” (*OR*), операція логічного множення “І” (*AND*) та операція “Складання, що виключає”, тобто “ЧИ” (*XOR*). Дані операції виконуються над бітами і особливості їх виконання наведені в таблиці 1.2.

У мікропроцесора є вісім команд, що виконують зсув 8-, 16-, 32- чи 64-бітового вмісту регістрів вправо чи вліво. Чотири з них зсувають число лінійно (лінійний зсув), а інші чотири його обертають його або циклічно зсувають (циклічний зсув). Є логічні та арифметичні команди лінійного зсуву. Логічні команди зсувають число не зважаючи на його знак. Вони використовуються для дій над числами без знаку та над нечисловими значеннями, наприклад, над масками. Арифметичні команди зберігають старший, тобто знаковий біт числа. Вони використовуються для дій над числами зі знаком.

Таблиця 1. 2 – Побітові логічні операції

Операція	Опис	Біт 1	Біт 2	Біт результату
NOT	Інвертування	0		1
		1		0
AND	Множення	0	0	0
		0	1	0
		1	0	0
		1	1	1
OR	Складання	0	0	0
		0	1	1
		1	0	1
		1	1	1
XOR	Складання, що виключає	0	0	0
		0	1	1
		1	0	1
		1	1	0

При лінійному логічному зсуві вліво (*SHL*) та лінійному арифметичному зсуві вліво (*SAL*) нуль заноситься у вакантний молодший біт. При лінійному логічному зсуві вправо (*SHR*) та лінійному арифметичному зсуві вправо (*SAR*) значення знакового розряду заноситься у вакантний старший біт. Біт, що вийшов за межі числа, вважається загубленим.

При циклічному зсуві вліво *ROL* або вправо *ROR* біт, що вийшов за межі числа, входить в нього з протилежного кінця. Одночасно цей біт відтворюється в прапорі *CF* регістра прапорів процесора. При циклічному зсуві вліво *RCL* та циклічному зсуві вправо *RCR* значення біта, що вийшов за межі числа з одного боку, потрапляє в прапор *CF*, а попереднє значення прапора *CF* заноситься до регістру з протилежного боку.

Приклад 2.

Визначити 3-й біт числа $D_{10} = 38$. Для визначення 3-го біта накладаємо маску, яка в 3-му біті має значення 1, а в інших бітах – 0, та виконуємо операцію логічного множення *AND*.

Номери бітів	7	6	5	4	3	2	1	0
Вага розрядів	128	64	32	16	8	4	2	1
D = 38	0	0	1	0	0	1	1	0
Логічна операція	*AND*	*AND*	*AND*	*AND*	*AND*	*AND*	*AND*	*AND*
Маска	0	0	0	0	1	0	0	0
Результат	0	0	0	0	0	0	0	0

Результат операції логічного множення в даному випадку дорівнює 0, значить третій біт числа *D* дорівнює 0.

Приклад 3.

Інвертувати 5-й біт числа D_{10}= 38. Для інвертування 5-го біта накладаємо маску, яка в 5-му біті має значення 1, а в інших бітах – 0, та виконуємо операцію *XOR*.

Номери бітів	7	6	5	4	3	2	1	0
Вага розрядів	128	64	32	16	8	4	2	1
D = 38	0	0	1	0	0	1	1	0
Логічна операція	*XOR*	*XOR*	*XOR*	*XOR*	*XOR*	*XOR*	*XOR*	*XOR*
Маска	0	0	1	0	0	0	0	0
Результат	0	0	0	0	0	1	1	0

У результаті накладання маски 5-й біт інвертовано, значення інших бітів не змінилися.

Приклад 4.

Встановити 4-й біт числа D_{10} = 38 в 1. Для цього накладаємо маску, яка в 4-му біті має значення 1, а в інших бітах – 0, та виконуємо операцію логічного складання *OR*.

Номери бітів	7	6	5	4	3	2	1	0
Вага розрядів	128	64	32	16	8	4	2	1
D = 38	0	0	1	0	0	1	1	0
Логічна операція	*OR*	*OR*	*OR*	*OR*	*OR*	*OR*	*OR*	*OR*
Маска	0	0	0	1	0	0	0	0
Результат	0	0	1	1	0	1	1	0

У результаті накладання маски 4-й біт встановлено в 1, значення інших бітів не змінилися.

Приклад 5.

Встановити значення 1-го біту числа $D_{10} = 38$ в 0. Щоб значення 1-го біту числа в 0, накладаємо маску, яка в 1-му біті має 0, а в інших бітах – 1, та виконуємо операцію логічного множення *AND*.

Номери бітів	7	6	5	4	3	2	1	0
Вага розрядів	128	64	32	16	8	4	2	1
$D = 38$	0	0	1	0	0	1	1	0
Логічна операція	*AND*	*AND*	*AND*	*AND*	*AND*	*AND*	*AND*	*AND*
Маска	1	1	1	1	1	1	0	1
Результат	0	0	1	0	0	1	0	0

У результаті накладання маски 1-й біт встановлено в нуль, значення інших бітів не змінилися.

Приклад 6.

Виконати лінійний логічний зсув числа $D_{10} = 38$ на 2 розряди вліво.

Номери бітів	7	6	5	4	3	2	1	0
Вага розрядів	128	64	32	16	8	4	2	1
$D = 38$	0	0	1	0	0	1	1	0
Операція зсуву	*SHL*	*SHL*	*SHL*	*SHL*	*SHL*	*SHL*	*SHL*	*SHL*
Результат	1	0	0	1	1	0	0	0

Приклад 7.

Виконати циклічний зсув *RCR* числа $D_{10} = 38$ на 2 розряди вправо. Значення прапора $CF = 1$.

Номери бітів	7	6	5	4	3	2	1	0
Вага розрядів	128	64	32	16	8	4	2	1
$D = 38$	0	0	1	0	0	1	1	0
Операція зсуву	*RCR*	*RCR*	*RCR*	*RCR*	*RCR*	*RCR*	*RCR*	*RCR*
Результат	0	1	0	0	1	0	0	1

Значення прапора *CF* після першого зсуву стало рівним 0, а після другого зсуву – знову 1.

1.4. Подання числової інформації в комп'ютерах

В комп'ютері використовуються три види чисел: з фіксованою комою, з плаваючою комою та двійково-десяткове подання. Крапка (кома) відділяє цілу частину числа від дробової. У чисел з фіксованою комою крапка розміщується або перед першою значущою цифрою (число менше нуля), або після останньої значущої цифри (число більше нуля). Перед старшим розрядом числа фіксується знак. Позитивні числа мають нульове значення знакового розряду, негативні – одиничне.

Числа з плаваючою комою мають різні формати подання, які, наприклад, містять: знак порядку, порядок, знак мантиси, мантису. У сучасних комп'ютерах формат поданням дійсних чисел наступна: знак числа, зміщений порядок (характеристика), мантиса. Порядок знаходиться як сума істинного порядку і деякої позитивної константи, яка називається зміщенням. Значення зміщення залежить від точності (наприклад, для одинарної точності воно дорівнює 127). Для подання числа одинарної точності (4 байти) у процесорах фірми Intel відводиться 1 біт – під знак, 8 біт – під порядок і 23 біта – під мантису.

Кількість біт, що відводять під кожне з полів, залежить від точності подання (одинарна, подвійна або розширена) та від типу процесора.

1.5 Машинні коди

У всіх комп'ютері без винятку всі операції виконуються над числами, які представлені спеціальними машинними кодами. Їхнє використання дозволяє обробляти знакові розряди чисел так само, як і значущі їх розряди, а також заміняти операцію віднімання операцією додавання.

Розрізняють **прямий код** (ПК), **зворотний код** (ЗК) і **додатковий код** (ДК) двійкового числа.

Прямий код позитивного двійкового числа збігається з його звичайним поданням у природній формі, тому що знак кодується нулем. Прямим кодом негативного числа називається його подання в природній формі запису, у якого в знаковому розряді ставиться 1. Такий код для виконання операції алгебраїчного додавання не застосовується, але він зручний для виконання операції множення і ділення.

Зворотний код позитивного двійкового числа збігається з його звичайним поданням у природній формі, тому що знак кодується нулем. Зворотний код негативного числа утворюється за наступним правилом: у знаковому розряді проставляється 1, а всі інші цифри в розрядах змінюються на зворотні. Зворотний код зручний для виконання операції алгебраїчного додавання, якщо врахувати простоту перекладу негативних чисел із прямого коду в зворотний. Однак виконувати операції множення і ділення в зворотному коді немає рації.

Перехід від зворотного коду числа до його природного поданням проводиться аналогічно.

Позитивні двійкові числа в додатковому коді подаються звичайним чином. Для одержання додаткового коду негативного числа необхідно в знаковому розряді записати 1, усі інформаційні розряди числа інвертувати і додати 1 до молодшого розряду. Зворотний перехід від додаткового коду числа до його звичайного подання проводиться аналогічно.

Приклад 8.

Подати числа $D = 38_{10}$ і $C = 17_{10}$, а також *(–D)* і *(–C)* у прямому, зворотному і додатковому кодах. Результат оформити у вигляді таблиці 3.

Таблиця 1. 3 – Машинні коди

	D	*C*	*–D*	*–C*
Прямий код	00100110	0 0010001	1 0100110	1 0010001
Зворотний код	00100110	0 0010001	1 1011001	1 1101110
Додатковий код	00100110	0 0010001	1 1011010	1 1101111

1.6 Арифметичні дії над числами

Усі сучасні комп’ютери мають досить розвинуту систему команд, що включає десятки і сотні машинних операцій. Однак виконання будь-якої операції засновано на використанні найпростіших мікрооперацій типу додавання і зсуву. Це дозволяє мати єдиний арифметико-логічний пристрій для виконання будь-яких операцій, які пов'язані з обробкою інформації. Правила додавання двійкових цифр двох чисел A і B представлені в таблиці 1.4.

Таблиця 1.4 – Правила додавання двійкових цифр

Значення i-х розрядів двійкових чисел A, B і переносу з попереднього розряду P_{i-1}			Розряд суми S_i	Перенос в наступний розряд Pi
a_i	b_i	P_{i-1}		
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Тут показані правила додавання двійкових цифр a_i і b_i однойменних розрядів з урахуванням можливих переносів з попереднього розряду P_{i-1}.

Операція додавання виконується в зворотному та додатковому кодах. Операція віднімання замінюється операцією додавання з від'ємним числом.

При виконанні операції додавання можливе переповнення розрядної сітки. Ознакою переповнення є відмінність знака отриманої суми від однакових знаків чисел, що складаються, тобто при додаванні двох позитивних чи двох негативних чисел. Для виявлення переповнення використовують *модифікований код*. У модифікованому коді під знак виділяються дві позиції. Знак позитивного числа кодується як 00, а негативного – як 11. Ознакою переповнення в модифікованому коді є відмінність цифр у знакових розрядах. Для усунення переповнення виконують нормалізацію мантиси.

Основні правила додавання двійкових чисел можна сформулювати так:

1. Операція віднімання змінюється операцією додавання з негативним числом.

2. Доданки повинні мати *однакове число розрядів*. Для вирівнювання розрядної сітки доданків можна дописувати незначущі нулі ліворуч для цілої частини числа і незначущі нулі праворуч для дробової частини числа.

3. Знакові розряди чисел беруть участь у додаванні так само, як і значущі.

4. Необхідні перетворення кодів виконуються зі зміною знаків чисел. Приписані незначущі нулі змінюють своє значення при перетвореннях за загальним правилом.

5. При утворенні одиниці переносу зі старшого знакового розряду, у випадку використання зворотного коду (ЗК) , ця одиниця *складається з молодшим числовим розрядом*. При використанні додаткового (ДК) одиниця переносу губиться. Знак результату формується автоматично, результат представляється в тому коді, у якому представлені вхідні доданки.

6. Знак суми виходить автоматично в процесі підсумовування вмісту знакових розрядів чисел і одиниці переносу, якщо вона є, із значущої частини числа.

7. При переповненні розрядної сітки проводять модифікацію суми. Ознакою переповнення в модифікованому коді є відмінність цифр у знакових розрядах. Для усунення переповнення число зсувають на один розряд вправо, а в правому знаковому розряді дублюють значення лівого знакового розряду.

Приклад 9

Над числами $D = 38_{10}$ і $C = 17_{10}$ виконати операції складання в модифікованих зворотному і додатковому кодах, використовуючи по черзі знаки "+" і "–" перед кожним з цих чисел, а саме : $+D+(+C)$, $+D+(–C)$, $–D+(+C)$, $–D+(–C)$.

Результат обчислень подати в прямому коді, а потім перевести в десятковий код та виконати перевірку.

Виконаємо операції в модифікованому зворотному коді.

Визначимо число $A = +D+(+C)$:

$D_{зк}$ = 00 100110

+

$C_{зк}$ = <u>00 010001</u>

$A_{зк}$ = 00 110111

Два нулі в знакових розрядах вказують на те, що число позитивне, прямий код співпадає зі зворотним кодом, тобто в 2-байтовому полі $A_{пр}$ = 00 00000000110111.

Зробимо перевірку, для цього число A переведемо в десятковий код:

$A_{10} = 2^0+2^1+2^2+2^4+2^5 = 55$

Визначимо число $A = + D +(–C)$:

$D_{зк}$ = 00 100110

+

$C_{зк}$ = 11 101110

$A_{зк}$ = 100 010100 ← переповнення розрядної сітки, старшу одиницю додаємо до молодшого розряду

$A_{зк}$ = 00 010101

Два нулі в знакових розрядах вказують на те, що число позитивне, прямий код співпадає зі зворотним кодом: $A_{пр}$ = 00 00000000010101.

Зробимо перевірку, для цього число A переведемо в десятковий код:

$A_{10} = 2^0+2^2+2^4 = 21$

Визначимо число $A = – D + (+C)$:

$D_{зк}$ = 11 011001

+

$C_{зк}$ = 00 010001

$A_{зк}$ = 11 101010

Дві одиниці в знакових розрядах вказують на те, що число негативне.

Для переводу в прямий код інвертуємо інформаційні розряди числа A:

$A_{пк}$ = 11 010101

Зробимо перевірку, для цього число A переведемо в десятковий код:

$A_{10} = –(2^0 + 2^2 + 2^4) = –21$

Визначимо число $A = – D + (– C)$:

$D_{зк}$ = 11 011001

+

$C_{зк}$ = $\underline{11\ 101110}$

$A_{зк}$ = 111 000111 ← переповнення розрядної сітки, старшу одиницю додаємо до молодшого розряду

$\overline{\ 11\ \ 001000\ }$

Дві одиниці в знакових розрядах вказують на те, що число негативне.

Для переводу в прямий код інвертуємо інформаційні розряди числа *A:*

$A_{пк}$ = 11 110111

Зробимо перевірку, для цього число *A* переведемо в десятковий код:

$A_{10} = -(2^0+2^1+2^2+2^4+2^5) = -55$

Виконаємо операції в модифікованому додатковому коді:

Визначимо число $A = +D+(+C)$:

$D_{дк}$ = 00 100110

+

$C_{дк}$ = $\underline{00\ 010001}$

$A_{дк}$ = 00 110111

Два нулі в знакових розрядах вказують на те, що число позитивне, прямий код співпадає з додатковим кодом, тобто $A_{пк}$ = 00 110111.

Зробимо перевірку, для цього число *A* переведемо в десятковий код:

$A_{10} = 2^0+2^1+2^2+2^4+2^5 = 55$

Визначимо число $A = +D+(-C)$:

$D_{дк}$ = 00 100110

+

$C_{д\,к}$ = $\underline{11\ 101111}$

$A_{дк}$ = 00 010101

Два нулі в знакових розрядах вказують на те, що число позитивне, прямий код співпадає з додатковим кодом, тобто $A_{пр}$ = 00 010101.

Зробимо перевірку, для цього число *A* переведемо в десятковий код:

$A_{10} = 2^0+2^2+2^4 = 21$

Визначимо число $A = (-D)+(+C)$:

$D_{дк}$ = 11 011010

+

$C_{дк}$ = $\underline{00\ 010001}$

$A_{дк}$ = 11 101011

Дві одиниці в знаковому розряді вказують на те, що число негативне. Для переходу в прямий код необхідно всі інформаційні розряди інвертувати і додати до молодшого розряду одиницю:

$$\begin{array}{r} 11\ 010100 \\ + \\ \underline{\qquad 1} \end{array}$$

$A_{пк}$ = 11 010101

Зробимо перевірку, для цього число *A* переведемо в десятковий код:

A_{10} = $-(2^{0+}2^2+2^4)$ = -21

Визначимо число $A = (-D)+(-C)$

$D_{дк}$ = 11 011010

+

$C_{дк}$ = $\underline{11\ 101111}$

$A_{дк}$ = 11 001001

Дві одиниці в знакових розрядах вказують на те, що число негативне. Для переходу в прямий код необхідно всі інформаційні розряди інвертувати і додати до молодшого розряду одиницю:

$$\begin{array}{r} 11\ 110110 \\ + \\ \underline{\qquad 1} \end{array}$$

$A_{пк}$ = 11 110111

Зробимо перевірку, для цього число *A* переведемо в десятковий код:

A_{10} = $-(2^0+2^1+2^2+2^4+2^5)$ = -5.

Приклад 10

Виконати операцію складання $A = 37{,}5$ та $B = 16{,}6$ в форматі з плаваючою комою. Точність двійкового подання A і B – п’ять знаків після коми. Очікуване десяткове значення суми – число 54,1.

Представимо $A_2 = 100101.10000$ та $B_2 = 10000.10011$ в форматі з плаваючою комою .Оскільки ці числа позитивні, то у знаковому розряді чисел A та B ставимо 0.

Ціла частина числа A займає 6 біт, тому порядок $p_a = 6_{10} = 110_2$.

Ціла частина числа B займає 5 біт, тому порядок $p_b = 5_{10} = 101_2$.

Знак порядків позитивний.

Мантиси чисел дорівнюють значенням інформаційних бітів цілої та дробової частин.

Таким чином, числа A і B у форматі з плаваючою комою виглядають так:

$A_{2пл} = 0\ 110\ 0\ 10010110000$

$B_{2пл} = 0\ 101\ 0\ 1000010011$

Порядки доданків різні, тому необхідно вирівняти розрядні сітки мантис. Для цього мантису числа з меншим порядком зрушуємо лінійно вправо на різницю порядків, а порядок цього числа збільшуємо на цю різницю. Для нашого випадку різниця порядків дорівнює 1, тому мантису числа B зрушуємо на 1 біт вправо, а порядок числа В збільшуємо на 1 і маємо:

$B_{2пл} = 0\ 110\ 0\ 01000010011.$

Записуємо значення мантис доданків у додатковому модифікованому коді:

$m_{a\ дк} = 00\ 10010110000 \quad m_{b\ дк} = 00\ 01000010011.$

Тепер порядки у доданків однакові і, отже, порядок суми дорівнює порядку доданків.

Складаємо значення мантис:

$m_{a\ дк} = 00\ 10010110000$

$+$

$\underline{m_{b\ дк} = 00\ 01000010011}$

$m_{f\ дк} = 00\ 11011000011$

$m_{f\ дк} = 00\ 11011000011$

Нулі у знаковому розряді результату вказують на відсутність переповнення розрядної сітки та на те, що число позитивне .

Запишемо значення суми в форматі з плаваючою комою:

$F_{2пл}$ = 0 110 0 11011000011

Таким чином у форматі з фіксованою комою, маємо F_2 = 110110.00011.

Зробимо перевірку, для цього переведемо число F_2 в десятковий код:

$F_{10} = 1*2^5+1*2^4+0*2^3+1*2^2+1*2^1+0*2^0+0*2^{-1}+0*2^{-2}+0*2^{-3}+1*2^{-4}+1*2^{-5}$ =

32+16+4+2+0,0625+0.0313 = 54.09375.

Оскільки під дробову частину було відведено тільки 5 біт, наш результат містить похибку. Відносна похибка складає приблизно 6 %.

2 КОНТРОЛЬНА РОБОТА 2

2.1 Завдання до контрольної роботи 2

Завдання до контрольної роботи 1 містить 3 пункти та наведено нижче.

1. Дано десяткове число A

A = №+K+15, №< = 10

A = №+K, 10<№<20

A = №+K–10, №> = 20

де № – номер за списком в журналі групи, K – код спеціальності (1, 2 або 3).

Число $B = A – 5$.

Подати числа A і B у прямому і додатковому кодах.

2. Знайти добуток чисел: $A*B$, $(–A)*B$, $A*(–B)$, $(–A)*(–B)$.

Операцію множення виконувати в додатковому коді в форматі з фіксованою та плаваючою комою. Результат подати в прямому коді, потім перевести в двійковий код і виконати перевірку.

3. Знайти частку від ділення A/B. Операцію ділення виконувати з відновленням і без відновлення залишку в форматі з фіксованою комою. Результат подати в прямому коді, потім перевести в двійковий код і виконати перевірку.

2.2 Множення чисел

2.2.1 Множення цілих чисел

Множення цілих чисел виконується в *прямому* і *додатковому* кодах.

При множенні чисел у *прямому коді* знакові й інформаційні розряди обробляються роздільно. Для визначення знаку результату цифри, які записані у знакових розрядах чисел складаються.

При множенні чисел у додатковому коді, знак формується автоматично. Множене подається у форматі для якого кількість бітів його подання подвоюється (так як кількість бітів подання результату добутку буде вдвічі більшою). Множення чисел виконується шляхом покрокового додавання значення множеного, що зсуваються, і часткової суми при наявності ненульового біта відповідного розряду множника. Множення можна виконувати, починаючи з молодших або зі старших розрядів множника. При цьому можна зсувати суму часткових добутків або множене. Найбільш простою за апаратурними витратами є схема множення чисел, яка виконує множення починаючи з молодших розрядів множника зі зсувом суми часткових добутків. Для прискорення виконання операції множення не додають часткову суму, яка дорівнює нулю.

Якщо множник є негативним числом, то необхідна **корекція результату**. Для корекції результату при множенні множеного A на знакову одиницю множника B необхідно скласти не $A_{дк}$, а $-A_{дк}$. Якщо множене є негативним числом, то в складанні беруть участь одиниці, отримані за рахунок інвертування незначущих нулів множеного. Результат отримуємо в додатковому коді. Якщо результат є від'ємним числом, то для переведення його в прямий код необхідно результат інвертувати та додати 1 до молодшого розряду

Приклад 1

Знайти добуток $C = A*B$, де $A_{10} = 8$, $B_{10} = 6$.

$A_{дк} =$ **0** 1000 $B_{дк} =$ **0** 110. Знак 0 + 0 = 0.

Добуток:

$$
\begin{array}{r}
\mathbf{0}00001000 \\
\underline{\mathbf{0}110} \\
\mathbf{0}00000000 \\
\mathbf{0}00010000 \\
\underline{\mathbf{0}00100000} \\
\mathbf{0}00110000
\end{array}
$$

Виконаємо перевірку: $C = 11000_2 = 1*2^5+1*2^4+0*2^3+0*2^2+0*2^1+0*2^0 = 48_{10}$

При множенні чисел у додатковому коді знак результату виходить автоматично.

Множення двох позитивних чисел у додатковому коді аналогічно множенню чисел у прямому коді.

Приклад 2

Знайти добуток $C = A*B$ в додатковому коді, де $A = 4$, $B = 7$.

A_{10} = 4, A_2 = 100, A $_{дк}$ = 0 100, B_{10} = 7, B_2 = 111, B $_{дк}$ = 0 111.

0000100
<u>   0111</u>
0000100
0001000
0010000
<u>**0**000000</u>
0111000

C $_{дк}$ = 0 11100. Так як отримане число більше 0, то додатковий код співпадає з прямим. Виконаємо перевірку:

C = 11100_2 = $1*2^4+1*2^3+1*2^2+0*2^1+0*2^0$ = 16+8+4 = 28_{10}

Приклад 3

Знайти добуток *C = A*B , D = A*(–B), E = (–A)*B, F = (–A)*(–B)*

A_{10} = 10, A_2 = 1010, A $_{дк}$ = 0 1010, – A $_{дк}$ = 1 0110.

B_{10} = 13, B_2 = 1101, B $_{дк}$ = 0 1101, – B $_{дк}$ = 1 0011.

Знайдемо *C* = *A*B:*

000001010
<u>   01101</u>
000001010
000000000
000101000
<u>**0**01010000</u>
010000010

C $_{дк}$ = 0 10000010. Оскільки отримане число більше 0, то додатковий код співпадає з прямим. Виконаємо перевірку:

C = 10000010_2 = $1*2^7+0*2^6+0*2^5+0*2^4+0*2^3+0*2^2+1*2^1+0*2^0$ = 128+2 = 130_{10}.

Знайдемо *D* = *A*(–B):*

000001010
<u>10011</u>
000001010 Корекція
000010100
000000000
000000000
<u>101100000</u>
101111110

При множенні на знакову одиницю множника була виконана корекція – складено $-A_{дк}$ = 1 01100000.

Оскільки добуток менше нуля, то для переходу в прямий код добуток інвертуємо та прибавляємо 1 до молодшого розряду: 1 10000001+1 = 1 10000010

Виконаємо перевірку:

D = 1 10000010_2 = $-(1*2^7+0*2^6+0*2^5+0*2^4+0*2^3+0*2^2+1*2^1+0*2^0)$ = $-(128+2)$ = -130_{10}.

Знайдемо $E = (-A)*B$:

111110110
<u>01101</u>
111110110
000000000
111011000
110110000
<u>000000000</u>
101111110

Оскільки множене є від'ємним числом, то при складанні беруть участь одиниці множеного (отримані за рахунок інверсії незначущих нулів) при наявності ненульового біта множника.

Оскільки добуток менше нуля, то для переходу в прямий код добуток інвертуємо та прибавляємо 1 до молодшого розряду: 1 10000001+1 = 1 10000010

Виконаємо перевірку:

$E = 1\ 10000010_2 = -(1*2^7+0*2^6+0*2^5+0*2^4+0*2^3+0*2^2+1*2^1+0*2^0) = -(128+2) = -130_{10}$.

Знайдемо $F = (-A) * (-B)$:

111110110

10011

111110110 Корекція

111101100

000000000

000000000

010100000

010000010

Оскільки множене є від’ємним числом, то при складанні беруть участь одиниці множеного (отримані за рахунок інверсії незначущих нулів) при наявності ненульового біта множника. При множенні на знакову одиницю множника виконується корекція, а саме додаємо - $(-A_{дк}) = +A_{дк} = 0\ 1010$.

$F_{дк} = 0\ 10000010$. Оскільки отримане число більше 0, то додатковий код співпадає з прямим. Виконаємо перевірку:

$F = 0\ 10000010_2 = 1*2^7+0*2^6+0*2^5+0*2^4+0*2^3+0*2^2+1*2^1+0*2^0 = 128 + 2 = 130_{10}$.

2.2.2. Множення дійсних чисел

При множенні дійсних чисел співмножники представляються в форматі з плаваючою комою.

Порядок добутку визначається шляхом складання порядків співмножників.

Добуток визначається множенням мантис співмножників, за правилами множення чисел з фіксованою комою у додатковому коді.

Якщо старший біт мантиси добутку після представлення його у прямому коді відмінний від одиниці, то виконують нормалізацію мантиси. Для нормалізації мантиси виконують лінійний зсув мантиси вліво на 1 розряд і зменшують порядок добутку на одиницю. Процедуру повторюють до появи в старшому біті мантиси одиниці.

Приклад 4

Подати числа $A = 3{,}5$ і $B = 12$ в форматі з плаваючою комою та знайти добуток $C = A*B$.

$A_{2пл} =$ 0 10 0 111, $B_{2пл} =$ 0 100 0 1100.

Знаходимо порядок добутку. Для цього складаємо порядки співмножників:

10+100 = 110.

Перемножуємо мантиси: $m_a =$ 000000111, $m_b =$ 0 1100.

$$
\begin{array}{r}
\mathbf{0}00000111 \\
\underline{01100} \\
\mathbf{0}00000000 \\
\mathbf{0}00000000 \\
\mathbf{0}00011100 \\
\mathbf{0}00111000 \\
\underline{\mathbf{0}00000000} \\
\mathbf{0}01010100
\end{array}
$$

$C_{2пл} =$ 0 110 0 1010100. Напишемо C_2 в форматі з фіксованою комою та виконаємо перевірку: $C_2 =$ 101010.0. $C_{10} = 1*2^5+0*2^4+1*2^3+0*2^2+1*2^1+0*2^0 = 42_{10}$.

2.3 Ділення чисел

2.3.1 Ділення чисел у додатковому коді в форматі з фіксованою комою

Розрізняють різні схеми ділення. Найбільш популярними є схеми з відновленням і без відновлення залишку. Операцію ділення виконують за допомогою операцій зсуву і додавання наступним чином.

Нехай *A* – ділене, *B* – дільник, *C* – частка (спочатку дорівнює нулю).

Представляємо ділене та дільник в додатковому коді та при необхідності врівноважуємо розрядну сітку (для цього дописуємо незначущі нулі зліва до числа з меншою кількістю біт).

Формування цілої частини частки *C* зводиться до операції віднімання від діленого *A* дільника *B* $(A = A–B)$ доти, доки різниця не стане меншою за нуль. Ціла частина частки буде дорівнювати кількості операцій віднімання із діленого A дільника B (при кожному відніманні значення цілої частини частки збільшується на одиницю $(C = C+1)$ за умови, якщо $A–B \varepsilon 0$).

Дробова частина частки формується за двома алгоритмами: з відновленням і без відновлення залишку. Кількість знаків після коми задається заздалегідь або залежить від форми подання чисел у комп'ютері.

Операцію ділення **без відновлення залишку (**за умови, що після формування цілої частини частки А $\bar{0}$) виконують так:

1. Виконати зсув A на один розряд вліво $(SHL\ A,1)$.
2. Якщо $A>0$, то знайти $A = A – B$, інакше знайти $A = A + B$.
3. Якщо $A\varepsilon\ 0$, то i-му розряду дробової частини присвоїти одиницю, інакше – нуль.
4. Якщо число знаків після коми менше необхідного і А $\bar{0}$, то повернутися до пункту 1, інакше закінчити формування дробової частини.

Операцію ділення **з відновленням залишку** (за умови, що після формування цілої частини частки А≠0) виконують так:

1. Виконати відновлення залишку $A = A+B$.
2. Виконати зсув A на однин розряд вліво $(SHL\ A,1)$.
3. Знайти різницю: $A = A–B$.
4. Якщо $A<0$, то i-му розряду дробової частини присвоїти нуль і виконати відновлення залишку: $A = A+B$, інакше i-му розряду дробової частини присвоїти одиницю.
5. Якщо число знаків після коми менше необхідного і А≠0, то повернутися до пункту 1, інакше закінчити формування дробової частини.

Приклад 5.

$A_{10} = 10$, $B_{10} = 3$. Знайти частку $C = A/B$ за алгоритмом без відновлення залишку.

$A_{\text{дк}} = 0\ 1010$, $B_{\text{дк}} = 0\ 0011$, $–B_{\text{дк}} = 1\ 1101$.

1) Знаходимо різницю $A = A–B$.

0 1010

<u>1 1101</u>

00111 >0 $C_2 = 1$ (різниця > 0, тому цілу частину збільшуємо на одиницю)

2) Знаходимо різницю $A = A–B$.

0 0111

<u>1 1101</u>

0 0100 >0 C_2 = 10 (різниця > 0, тому цілу частину збільшуємо на одиницю)

3) Знаходимо різницю $A = A–B$.

0 1000

<u>1 1101</u>

0 0001 >0 C_2 = 11 (різниця > 0, тому цілу частину збільшуємо на одиницю)

4) Знаходимо різницю $A = A–B$.

0 0001

<u>1 1101</u>

1 1110 <0 C_2 = 11, (різниця < 0 – формування цілої частини завершено)

5) Виконуємо зсув числа 1 1110, отримуємо число 1 1100

6) Виконуємо $A = A+B$ (оскільки $A < 0$).

1 1100

<u>0 0011</u>

1 1111 <0 C_2 = 11,0 (сума < 0, тому i-му розряду дробової частини присвоюємо нуль)

7) Виконуємо зсув числа 1 1111, отримуємо число 1 1110

8) Знаходимо суму $A = A+B$.

1 1110

<u>0 0011</u>

0 0001 >0 C_2 = 11,01 (сума > 0, тому i-му розряду присвоюємо одиницю)

9) Виконуємо зсув числа 0 0001, отримуємо 0 0010

10) Знаходимо різницю $A = A–B$.

0 0010

<u>1 1101</u>

1 1111 <0 C_2 = 11,010... (сума < 0, тому i-му розряду присвоюємо нуль)

11) Виконуємо перевірку результату.

$1\ 0\ \frac{1}{2}\ \frac{1}{4}\ \frac{1}{8}$

$11{,}010_2 \longrightarrow (3+0{,}25)_{10} = 3{,}25_{10}$

Приклад 6.

$A_{10} = 19$, $B_{10} = 14$. Знайти частку $C = A/B$ за алгоритмом без відновлення залишку.

$A_{дк}$ = 0 10011 $B_{дк}$ = 0 1110 $-B_{дк}$ = 1 0010.

1) Знаходимо різницю $A = A–B$.

0 10011

1 10010

0 00101 >0 C_2 = 1 (різниця > 0, тому цілу частину збільшуємо на одиницю)

2) Знаходимо різницю $A = A–B$.

0 00101

1 10010

1 10111 <0 C_2 = 1, (різниця < 0 – формування цілої частини завершено)

3) Виконуємо зсув числа 1 10111, отримуємо число 1 01110

4) Виконуємо $A = A+B$ (оскільки $A < 0$).

1 01110

0 01110

1 11100 <0, тому i-му розряду присвоїти нуль, C_2 = 1,0

5) Виконуємо зсув числа 1 11100, отримуємо число 1 11000

6) Знаходимо суму $A = A+B$.

1 11000

0 01110

0 00110 >0, тому i-му розряду присвоїти одиницю, C_2 = 1,01

7) Виконуємо зсув числа 0 00110, отримуємо число 0 01100

8) Оскільки A>0, то знаходимо різницю $A = A–B$.

0 01100

<u>1 10010</u>

11110 <0, тому i-му розряду присвоїти нуль, C_2 = 1,010

9) Виконуємо зсув числа 1 11110, отримуємо число 1 1110

10) Виконуємо $A = A+B$ (оскільки $A < 0$).

1 11100

0<u> 01110</u>

0 01010 >0,тому i-му розряду присвоїти одиницю, C_2 = 1,0101...

11) Виконуємо перевірку результату.

$$\overset{0}{1},\overset{\frac{1}{2}}{0}\overset{\frac{1}{4}}{1}\overset{\frac{1}{8}}{0}1_2 \longrightarrow (1+0,25+0,0625)_{10} = 1,3125_{10}.$$

Приклад 7.

A_{10} = 12, B_{10} = 5. Знайти частку $C = A / B$ за алгоритмом з відновленням залишку.

A $_{дк}$ = 0 1100 B $_{дк}$ = 0 0101 $-B$ $_{дк}$ = 1 1011

1) Знаходимо різницю $A = A - B$.

0 1100

<u>1 1011</u>

0 0111 >0 C_2 = 1 (різниця > 0, тому цілу частину збільшуємо на одиницю)

2) Знаходимо різницю $A = A–B$.

0 0111

<u>1 1011</u>

0 0010 >0 C_2 = 10 (різниця > 0, тому цілу частину збільшуємо на одиницю)

3) Знаходимо різницю $A = A–B$.

0 0010

<u>1 1011</u>

1 1101 <0 C_2 = 10, (різниця < 0 – формування цілої частини завершено)

4) Виконуємо відновлення залишку.

1 1101

0 0101

0 0010.

5) Виконуємо зсув 0 0100

6) Знаходимо різницю $A = A–B$.

0 0100

1 1011

1 1111 <0 , тому i-му розряду присвоїти нуль, C_2 = 10,0

7) Виконуємо відновлення залишку.

1 1111

0 0101

0 0100.

8) Виконуємо зсув: 0 1000.

9) Знаходимо різницю $A = A–B$.

0 1000

1 1011

0 0011 >0, тому i-му розряду присвоїти одиницю, C_2 = 10,01...

10) Виконуємо перевірку результату.

1 0 $\frac{1}{2}$ $\frac{1}{4}$

$10{,}01_2$ —> $(2+0{,}25)_{10}$ = $2{,}25_{10}$.

Приклад 8.

A = 19, B = 14. Знайти частку $C = A/B$ по схемі з відновленням залишку.

$A_{дк}$ = 0 10011 $B_{дк}$ = 0 01110 $–B_{дк}$ = 1 10010

1) Знаходимо різницю $A = A–B$.

0 10011

1 10010

0 00101 >0 $C_2 = 1$ (різниця > 0, тому цілу частину збільшуємо на одиницю)

2) Знаходимо різницю $A = A–B$.

0 00101

1 10010

1 10111 <0 $C_2 = 1$, (різниця < 0 – формування цілої частини завершено)

3) Виконуємо відновлення залишку.

1 10111

0 01110

0 00101

4) Виконуємо зсув: 0 01010.

5) Знаходимо різницю $A = A–B$.

0 01010

1 10010

1 11100 <0, тому i-му розряду присвоїти нуль, $C_2 = 1{,}0$.

6) Виконуємо відновлення залишку.

1 11100

0 01110

0 01010.

7) Виконуємо зсув: 0 10100.

8) Знаходимо різницю $A = A–B$.

0 10100

1 10010

0 00110 >0, тому i-му розряду присвоїти одиницю, $C_2 = 1{,}01$

9) Виконуємо зсув: 0 01100.

10) Знаходимо різницю $A = A–B$.

0 01100

1 10010

1 11110 <0, тому i-му розряду присвоїти нуль, $C_2 = 1{,}010...$

11) Виконуємо перевірку результату.

$0 \ \frac{1}{2} \ \frac{1}{4} \ \frac{1}{8}$

$1{,}010_2 \longrightarrow (1+0{,}25)_{10} = 1{,}25_{10}$.

3 РЕКОМЕНДАЦІЇ ЩОДО ОФОРМЛЕННЯ ЗАВДАННЯ

Завдання виконується українською мовою в зошитах або на аркушах формату А4, які потім зшиваються в окремий документ. Оформлене завдання підписується наступним чином:

Контрольна робота
з курсу Основи комп'ютерної інженерії
Виконав студент групи ....
Прізвище, ім'я, по батькові.

При виконанні завдань відповідно до кожного пункту необхідно виконати перевірку, а саме – перевести кожен результат спочатку в прямий код, а потім в десяткову систему числення.

Рекомендована література

1. Гавриленко С.Ю, Клименко А.М.,. Гоготов В.В «Основи комп'ютерної техніки». – Харків, НТУ «ХПІ», 2008, – 272 с.
2. Гавриленко С.Ю., Клименко А.М. Методичні вказівки до виконання розрахункового завдання за курсом "Організація роботи комп'ютера" Харків: НТУ "ХПІ", 2007. – 36 с
3. Гавриленко С.Ю., Клименко А.М. Комп'ютерна логіка. Лабораторний практикум. Х.: НТУ «ХПІ», 2015. – 75 с.
4. Гавриленко С.Ю., Клименко А.М. Любченко Н.Ю. Теорія цифрових автоматів та формальних мов. Рекомендовано МОН України – Харків, НТУ «ХПІ», 2010. – 176 с..
5. Гавриленко С.Ю., Клименко А.М. Носков. В.І Логіка дискретних автоматів: навч.посіб.–Х: НТУ «ХПІ», 2014.–129 с.
6. Gavrylenko S. The basis of architecture computer systems / Gavrylenko S., Khatsko N. – Kharkiv : NTU "KhPI", 2019. – 75 p.
7. Кравчук С.О., Шонін В.О. Основи комп'ютерної техніки. – Киев: Політехніка, 2005. – 344 с.
8. Матвієнко М.П. Комп'ютерна логіка. Навчальний посібник. - Київ: ТОВ "Центр навчальної літератури", 2012. – 288 с.

Навчальне видання

ГАВРИЛЕНКО Світлана Юріївна,
ОКТЯБРЬОВА Олена Володимирівна

Методичні вказівки

до виконання контрольних робіт за курсом “ Основи комп’ютерної інженерії” для студентів заочної форми навчання спеціальності “Комп’ютерна інженерія

Роботу до видання рекомендувала Філоненко А.М

В авторській редакції

Самостійне електронне видання